

# **CAHIER DES CHARGES SNIFFR**

<b>1. Introduction.....</b>	<b>1</b>
1.1. Contexte du projet.....	1
1.2. Objectifs du projet.....	2
<b>2. Glossaire.....</b>	<b>2</b>
<b>3. Description générale.....</b>	<b>2</b>
3.1. Vue d'ensemble du système.....	2
3.2. Besoin utilisateur.....	2
3.3. Contraintes générales.....	3
<b>4. Exigences fonctionnelles.....</b>	<b>3</b>
4.1. Fonctionnalités principales.....	3
4.2. Cas d'utilisation.....	3
4.3. Scénarios d'utilisation.....	3
<b>5. Exigences non fonctionnelles.....</b>	<b>3</b>
<b>6. Architecture technique.....</b>	<b>4</b>
6.1. Description Globale.....	4
6.2. Schémas et diagrammes.....	4
6.3. Technologies et outils.....	5
<b>7. Spécification détaillées.....</b>	<b>5</b>
7.1. Spécification des composants.....	5
7.2. Interface utilisateur.....	5
<b>8. Plan de tests et validations.....</b>	<b>6</b>
8.1. Stratégie de tests.....	6
8.2. Cas de test.....	6
8.3. Critères d'acceptation.....	7
<b>9. Plan de déploiement.....</b>	<b>7</b>
9.1. Stratégie de déploiement.....	7
9.2. Formation des utilisateurs.....	7
<b>10. Gestion de projet.....</b>	<b>7</b>
10.1. Planning.....	7
10.2. Rôles et responsabilités.....	8

# 1. Introduction

## 1.1. Contexte du projet

SniffR est une entreprise spécialisée dans les technologies connectées pour les animaux de compagnie. Dans le cadre du lancement de son premier produit, SniffR souhaite développer une solution complète permettant de suivre en temps quasi-réel la position géographique des animaux via un collier connecté.

## 1.2. Objectifs du projet

Offrir aux propriétaires d'animaux une solution de géolocalisation précise.

Permettre la consultation et l'analyse des données GPS via une plateforme web.

Garantir la sécurité et la fiabilité des données échangées entre les IoTs et la plateforme.

Fournir une interface conviviale pour la gestion des dispositifs.

# 2. Glossaire

Termes	Définition
IoT	Objet connecté, ici un module GPS intégré à un collier.
GNSS	Système de géolocalisation global, utilisé pour obtenir des données de localisation..
IMEI	Identifiant unique d'un module IoT (ex. : 457125485236021).
TCP	Protocole de communication réseau utilisé pour la transmission des données.
API	Interface de Programmation Applicative pour accéder aux données côté frontend.
Mapbox	Outils de cartographie pour visualiser les positions des IoTs.

### **3. Description générale**

#### **3.1. Vue d'ensemble du système**

Le système sera composé de différents IoT ESP32-C3 SIM (nombre variable en fonction de la demande client), d'un serveur TCP, d'une API Express exposant les données aux clients, d'une base de données et enfin d'une application web publique.

#### **3.2. Besoin utilisateur**

L'utilisateur doit pouvoir créer un compte, se connecter, ajouter de nouveaux IoTs. Accéder à la map de géolocalisation, visualiser l'historique des positions, voir des statistiques globales.

#### **3.3. Contraintes générales**

La plateforme web doit être accessible depuis tous lieux et les navigateurs Chrome, Firefox, Edge, Safari.

Le système de mappemonde doit fonctionner en temps réel (rafraîchissement < 2 minutes).

L'ensemble du système doit être scalable.

### **4. Exigences fonctionnelles**

#### **4.1. Fonctionnalités principales**

Réception et stockage des données IoT (GPS, batterie).

Authentification.

Gestion des IoTs (ajout, modification).

Visualisation des données GPS sur une carte.

Génération de graphiques de données.

#### **4.2. Cas d'utilisation**

L'utilisateur crée un compte, ajoute un IoT en saisissant son IMEI.

Un IoT envoie sa position GPS via TCP au serveur.

Le dashboard récupère les données GPS pour les afficher sur une carte.

L'utilisateur consulte la map avec les coordonnées GPS.

### 4.3. Scénarios d'utilisation

Un utilisateur ouvre la carte pour localiser son animal perdu.

Un utilisateur consulte le nombre de positions reçues au cours des 7 derniers jours pour s'assurer que son IoT fonctionne encore.

Un utilisateur modifie le nom d'un IoT dans sa liste pour mieux identifier son chien.

## 5. Exigences non fonctionnelles

Critères	Exigences
Performance	Support de 100+ IoTs envoyant 1 point/minute.
Fiabilité	99,9 % de disponibilité du backend. Redondance serveur possible.
Scalabilité	Base de données extensible sans refonte majeure.
Maintenabilité	Code modulaire et documenté. Utilisation de Mongoose/Express pour clarté.
Compatibilité	Application web accessible sur tous les navigateurs modernes.

## 6. Architecture technique

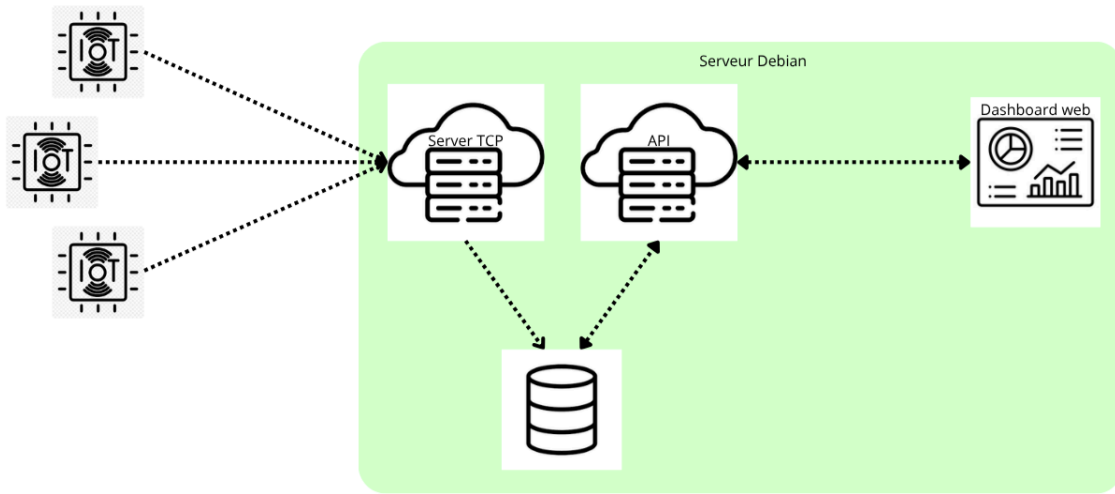
### 6.1. Description Globale

Le système repose sur une architecture en couches :

IoT → Serveur TCP → MongoDB

MongoDB → API REST → Dashboard Web (React)

## 6.2. Schémas et diagrammes



## 6.3. Technologies et outils

Composant	Technologie
Frontend	React, Mapbox, ApexCharts
Backend API	Node.js, Express.js, Jest, K6
Serveur TCP	Node.js, net (TCP natif), CBOR
DB	MongoDB (via Mongoose)
Hardware	Arduino Framework, nlohmann-json, CBOR

## 7. Spécification détaillées

### 7.1. Spécification des composants

Le serveur TCP doit permettre l'écoute des connexions entrantes, de réceptionner les messages bruts, de décoder et de valider les données reçues et de stocker les données GPS, l'état de l'IoT (batterie) ou autre capteur en base de données.

L'API doit faire la liaison entre la base de données et l'application web, son travail est donc de fournir les endpoints nécessaires aux opérations réalisables telles que : la connexion, la création de compte, la gestion des IoTs (création, modification), la consultation des données IoT (position GPS, batterie).

La base de données MongoDB quant à elle est chargée du stockage des données, qu'elles soient utilisateur, IoTs, données des IoTs ou relatives au group

## 7.2. Interface utilisateur

L'interface graphique doit avoir une interface sobre et moderne et contiendra : une page de connexion, une page de création de compte, une page "dashboard" exposant différentes statistiques et graphiques, une page de visualisation des IoTs, une page d'ajout des IoTs et une page mappemonde interactive.

## 8. Plan de tests et validations

### 8.1. Stratégie de tests

Test fonctionnel (sous Insomnia & Swagger) et test unitaire des routes API.

Test fonctionnel du système embarqué.

Test fonctionnel du site web.

Test de charge de 10, 50 et 100 requêtes en simultané pendant ~2 minutes.

### 8.2. Cas de test

Fonctionnalités	tests réalisés
Connexion	Connexion avec un utilisateur existant. Connexion avec un mauvais mot de passe. Connexion avec un email inexistant. Connexion sans email. Connexion sans mot de passe.
Création de compte	Inscription avec données valides. Inscription avec email déjà utilisé. Inscription avec mot de passe ne répondant pas aux exigences.
Déconnexion	Test de connexion avec un token de connexion valide. Test de déconnexion avec un token de connexion invalide.
Accès info utilisateur (/me)	Test avec un utilisateur connecté. Test sans utilisateur connecté.
Récupération des IoTs d'un utilisateur	Test avec un seul IoT relié à l'utilisateur. Test sans IoT relié à l'utilisateur. Test avec plusieurs IoTs reliés à l'utilisateur.
Modification des informations d'un IoT	Test avec information valide. Test avec un nom trop court.
Association d'un IoT à un utilisateur	Test avec information valide.

	Test avec un IoT déjà appairé. Test avec un IoT inexistant.
Donnée pour graphique nombre de data reçues par groupe	Test avec un groupe qui a des datas. Test avec un groupe sans data.
Donnée pour graphique historique de batterie	Test avec information valide. Test avec un IoT sans data.
Donnée GPS pour la map	Test avec des données valides. Test avec un IoT qui n'existe pas. Test avec un IoT sans data.

### 8.3. Critères d'acceptation

Tous les tests sur les fonctionnalités majeures doivent passer à 100%.

Les tests de charge doivent avoir en dessous de 10% d'erreur et 90% de réponse en dessous de 500ms.

## 9. Plan de déploiement

### 9.1. Stratégie de déploiement

Nous souhaitons commencer par un déploiement au niveau de la métropole lilloise avant de nous déployer dans les villes voisines puis dans l'hexagone. Cette première phase se fera sous forme d'itération progressive afin d'évaluer si les besoins des clients sont satisfaits et de corriger les différents bugs ou erreurs à chaud en hot reload.

### 9.2. Formation des utilisateurs

Afin de former les utilisateurs de notre système nous organiserons avec les clients 3 sessions de formation pour eux et leurs collaborateurs:

- Session 1. Présentation de la technologie (2 heures)
- Session 2. Explications avancées (5 heures)
- Session 3. Atelier pratique de la solution (3 heures)





## 10. Gestion de projet

### 10.1. Planning

Phase	Durée
Spécifications	1 semaines
Développement	8 semaines
Tests	2 semaines
Déploiement	1 semaines

### 10.2. Rôles et responsabilités

Rôles et responsabilités	
 Nom	 Rôle
Erwan Mayolle	Développeur ▾
stevenfardoux62670@gmail.com	Développeur ▾ Gestion de projets ▾